Mackly Monestime

```
L1        13126 S (ARRAY PROCESSOR# OR PARALLEL PROCES?)
L2       187500 S (MULTIPLE OR PLURALITY OR MORE THAN ONE) (2A) (?PROCESSOR OR
L3          262 S (SPLIT? OR DIVID?) (3A) DIMENS? (3A) SPAC?
L4           13 S L3 (P) (SUB-SPACE# OR SUBSPACE#)
L5       192610 S L1 OR L2
L6            0 S L5 (P) L4
L7            6 S L5 AND L4

=>
```

Mackly Monestime

L7 ANSWER 1 OF 6 USPATFULL

PI US 6272487 B1 20010807

DETD Another approach is to generate a new type of multi-column quantile statistic, 650 (FIG. 6A). one that **divides** n-**dimensional space** into smaller, bounded, n-dimensional **sub-spaces**, with each **sub-space** containing approximately the same number of tuples, 651 (FIG. 6A). For example, consider the distribution of tuples shown in FIG. 4.

DETD When using the dynamic **process**, only **one** pass of the ordered set of rows is needed. During that pass the desired number of rows in each quantile, and the low and high bounds of each quantile, are dynamically adjusted as needed so that each quantile contains approximately the same number of rows. This algorithm might result in the last quantile being larger or smaller than the rest, but it saves one scan of the ordered set of rows (if the total number of rows isn't known).

DETD FIG. 5B shows the context of the query engine 500 in a database management system (DBMS) 540 in a processing system 1 having memory 521 and at least one cpu 522. The system 1could be connected to other systems 2 via a network 530. The application 501 could be resident on any of the systems 1, 2, in the network or could be any user connected to any one of the systems via input/output user interface devices (e.g., keyboard, display, etc.). The system, method and program of this invention is applicable to any type of database management system whether it is contained within a single system or is within a networked environment including **parallel processing** systems, client/server processing systems, distributed systems, etc. Although the invention herein is described in reference to relational database management systems, multi-column statistics are applicable and adaptable to other database systems including object oriented systems. For example, the invention is easily adaptable to take into consideration a correlation and relationship among objects such as through multi-object statistics similar to the multi-column statistics described herein.

DETD A machine embodying the invention may involve **one** or more **processing** systems including, but not limited to, cpu, memory/storage devices, communication links, communication/transmitting devices, servers, I/O devices, or any subcomponents or individual parts of **one** or more **processing** systems, including software, firmware, hardware or any combination or subcombination thereof, which embody the invention as set forth in the claims.

L7 ANSWER 2 OF 6 USPATFULL

PI US 6223182 B1 20010424

DETD The invention is related to the use of computer system 100 for organizing data. According to one embodiment of the invention, organizing data is provided by computer system 100 in response to **processor** 104 executing **one** or more sequences of one or more instructions contained in main memory 106. Such instructions may be read into main memory 106 from another computer-readable medium, such as storage device 110. Execution of the sequences of instructions contained in main memory 106 causes processor 104 to perform the process steps described herein. **One** or more **processors** in a multi-processing arrangement may also be employed to execute the sequences of instructions contained in main memory 106. In alternative embodiments, hard-wired circuitry may be used in place of or in combination with software instructions to implement the invention. Thus, embodiments of the invention are not limited to any specific combination of hardware circuitry and software.

DETD BH codes model a regular, recursive decomposition of a space into a

plurality of **subspaces** formed by dividing each coordinate direction in half. A two-dimensional space is subdivided into four rectangular **subspaces,** commonly called a quadtree, and, an n-**dimensional space** is **divided** into 2.sub.n **subspaces**. Each **subspace** may be further subdivided into additional 2.sub.n **subspace** as necessary. Depending on the resolution desired, there is no theoretical limit to the number of decomposition levels available, but practically the number of levels is limited by the particular computer system utilized. A thirty-two level decomposition of the world's surface is capable of attaining a resolution of 9.3 mm.times.4.7 mm. Thus, BH codes indicate an orderly decomposition of space.

CLM What is claimed is:

10. A computer-readable medium bearing instruction for organizing data in a data container including a plurality of records, each of said plurality of records including a plurality of fields, said instructions arranged for causing **one** or more **processors** to perform the steps of: determining codes for corresponding records of said plurality records based on bit-interleaving values from at least two said plurality of fields belonging to said corresponding records; creating a first database object having a plurality of rows corresponding to said plurality of records and a first column for holding said codes for said corresponding records and a second column for holding a reference to said corresponding records; creating a second database object containing prefixes of said codes based on said first database object; and subdividing the data container into a plurality of subsets based on said second database object.

16. A computer-readable medium bearing instruction for organizing data in a data container including a plurality of records, each of said plurality of records including a plurality of fields, said instructions arranged for causing **one** or more **processors** to perform the steps of: determining codes for corresponding records of said plurality records based on bit-interleaving values from at least two of said plurality of fields belonging to said corresponding records; subdividing the data container into a plurality of subsets based on said codes; storing said plurality of subsets as a tree data structure having a plurality of entries corresponding to said plurality of records arranged in an order dictated by said codes for said corresponding records.

L7 ANSWER 3 OF 6 USPATFULL

PI US 5995957 19991130

DETD Another approach is to generate a new type of multi-column quantile statistic, 650 (FIG. 6A); one that **divides** n-**dimensional space** into smaller, bounded, n-dimensional **sub-spaces,** with each **sub-space** containing approximately the same number of tuples, 651 (FIG. 6A). For example, consider the distribution of tuples shown in FIG. 4.

DETD When using the dynamic **process,** only **one** pass of the ordered set of rows is needed. During that pass the desired number of rows in each quantile, and the low and high bounds of each quantile, are dynamically adjusted as needed so that each quantile contains approximately the same number of rows. This algorithm might result in the last quantile being larger or smaller than the rest, but it saves one scan of the ordered set of rows (if the total number of rows isn't known).

DETD FIG. 5B shows the context of the query engine 500 in a database management system (DBMS) 540 in a processing system 1 having memory 521

and at least one cpu 522. The system 1 could be connected to other systems 2 via a network 530. The application 501 could be resident on any of the systems 1, 2, in the network or could be any user connected to any one of the systems via input/output user interface devices (e.g., keyboard, display, etc.). The system, method and program of this invention is applicable to any type of database management system whether it is contained within a single system or is within a networked environment including **parallel processing** systems, client/server processing systems, distributed systems, etc. Although the invention herein is described in reference to relational database management systems, multi-column statistics are applicable and adaptable to other database systems including object oriented systems. For example, the invention is easily adaptable to take into consideration a correlation and relationship among objects such as through multi-object statistics similar to the multi-column statistics described herein.

DETD A machine embodying the invention may involve **one** or more **processing** systems including, but not limited to, cpu, memory/storage devices, communication links, communication/transmitting devices, servers, I/O devices, or any subcomponents or individual parts of **one** or more **processing** systems, including software, firmware, hardware or any combination or subcombination thereof, which embody the invention as set forth in the claims.

L7 ANSWER 4 OF 6 USPATFULL

PI US 5909681 19990601

TI Computer system and computerized method for partitioning data for **parallel processing**

AB A computer system splits a data space to partition data between processors or processes. The data space may be split into sub-regions which need not be orthogonal to the axes defined by the data space's parameters, using a decision tree. The decision tree can have neural networks in each of its non-terminal nodes that are trained on, and are used to partition, training data. Each terminal, or leaf, node can have a hidden layer neural network trained on the training data that reaches the terminal node. The training of the non-terminal nodes' neural networks can be performed on **one processor** and the training of the leaf nodes' neural networks can be run on separate processors. Different target values can be used for the training of the networks of different non-terminal nodes. The non-terminal node networks may be hidden layer neural networks. Each non-terminal node automatically may send a desired ratio of the training records it receives to each of its child nodes, so the leaf node networks each receives approximately the same number of training records. The system may automatically configures the tree to have a number of leaf nodes equal to the number of separate processors available to train leaf node networks. After the non-terminal and leaf node networks have been trained, the records of a large data base can be passed through the tree for classification or for estimation of certain parameter values.

SUMM In general, a major issue in parallel computing is the division of the computational task so that a reasonable percentage of the computing power of **multiple processor** can be taken advantage of and so the analytical power of the process is as high as possible. This issue is particularly important when it comes to many data base mining functions, such the training of neural networks mentioned above or of other modeling tasks.

SUMM According to one aspect of the present invention a computer system with P processors receives data objects having N parameters. It **divides** an N-**dimensional** data **space** defined by the N parameters into M **sub-spaces**, where M is

greater than or equal to P. This is done in such a manner that the boundaries between the resulting **sub-spaces** need not be orthogonal to the N-dimensions. The system associates a different set of one or more **sub-spaces** with each of the P processors. It distributes data objects located in each **sub-space** to the **sub-space**'s associated processor and causes each processor to perform a computational process on each of the data objects distributed to it.

SUMM According to another aspect of the invention, a computer system **divides** an N-**dimensional** data **space**, having a separate dimension for each of N parameters associated with the data set, into M **sub-spaces**. It associates each of these M **sub-spaces** with a corresponding one of M hidden-layer neural networks, and uses the data objects in each of the M **sub-spaces** to train that **sub-space**'s associated hidden-layer neural network. The resulting divisions need not be orthogonal to the N dimensions of the space.

SUMM According to another aspect of the invention, a computer system creates a decision tree having a neural network for each of its nodes, including a hidden-layer network for each of its terminal, or leaf, nodes. Each of the tree's non-terminal nodes use the portion of the training data which is supplied to it to train its associated neural network and then uses that neural network, once trained, to determining which of the training data object supplied to it should be supplied to each of its child nodes. In one embodiment, the net in each non-terminal node is trained to **divide** an N-**dimensional space** defined by parameters from the training data set into **sub-spaces**, and the data objects associated with each **sub-space** are routed to a different one of that non-terminal node's child nodes. In such an embodiment, each non-terminal node can be a two layer neural network which defines a single vector of weights in the N-dimensional space, and the data space is split by a plane perpendicular to that vector.

SUMM The portion of the training set supplied by the decision tree to each of its terminal, or leaf, nodes is used to train that node's corresponding neural network. In preferred embodiments, different leaf node networks are trained on different processors. In many embodiments, a copy of the entire decision tree, including the neural networks in both its non-terminal and leaf nodes, is stored on each of a **plurality** of **processors**. Then a set of new data objects is split into separate data partitions, one for each of such processor. Finally data objects from the partition associated with each processor are passed down through the copy of the complete decision tree stored on that processor. This causes each such data object to be routed to a given leaf node of the tree, at which point the hidden-layer neural network associated with the given leaf node will analyze the data object, such as by classifying it, or recording an estimated value for each of its target fields.

DRWD FIG. 3 illustrates BuildModel.sub.-- Master, a simplified pseudo-code representation of the **process** run on **one processor** to train the non-terminal nodes of the neural tree network as part of the training process shown in FIG. 2;

DRWD FIG. 13 illustrates BuildModel.sub.-- Slave, a simplified pseudo-code representation of the process run on each of a **plurality** of **processors** to train the hidden-layer neural networks associated with the leaf nodes of the neural tree network shown in FIG. 2;

Mackly Monestime

DRWD FIG. 17 illustrates ApplyModel.sub.-- Slave, a simplified pseudo-code representation of the process run on each of a **plurality** of separate **processor** nodes in the ApplyModel process shown in FIGS. 14 and 15;

DETD In FIG. 1, **one** of the **processor** nodes 52A is labeled a "master". The other of the processor nodes are labeled "slave". In the **parallel processing** scheme used in a preferred embodiment of the invention, certain computational processes are best performed on one machine. Also there is a benefit in having one machine tell the others what to do. This one machine is called the Master, since it controls the operation of other, slave, processors. In the embodiment shown in the figures, the master runs on a different machine than any of the slaves. In other embodiments, a single processor can act as both a master and a slave.

DETD In the embodiment of the invention described, the ApplyModel **process** is **one** of a set of modular computing processes 180 which can be run on a parallel computer. If the ApplyModel process 180A is being run without any preceding modular process, as shown schematically in FIG. 18, or with an immediately preceding modular process which does not produce a separate partition for each of the processors to be used in the ApplyModel process, the partitioning process 182 which is part of the module 180A will have to partition the apply data base, as indicated in step 172.

DETD The neural tree network produced by the above method has the advantage of performing better analysis for a given level of computation than prior neural networks or prior neural tree networks. By **dividing** the N-**dimensional** data **space** into **sub-spaces** and using each such **sub-space** to train a separate end-node hidden-layer neural network, the distribution of training samples fed to each such end net are much more similar. This results in three advantages: 1) it takes fewer hidden-layer nodes to accurately model the data supplied to each network: 2) it takes fewer training cycles to train each hidden-layer networks; and 3) each training cycle has fewer training records. Each of these three factors alone results in computational savings. Their combination results in a much greater one.

DETD The use of such hidden-layer neural networks has the effect of recursively **splitting** the N-**dimensional** **space** defined by the records of the training set into **sub-spaces**, as does the embodiment of the invention using two layer nets. The difference is that the boundaries of the **sub-spaces** created with hidden-layer nets in the non-terminal tree nodes of FIG. 20 are curved in N-dimensional space, allowing for a division of records between leaf nodes which is more likely to group together into a common leaf node records which are similar for purposes of the analysis task. This further improves the accuracy of the neural tree network's analysis.

DETD Similarly, the neural tree network processes described above could all be run on **one processor**. Or if run on **multiple processors**, they could be run on **multiple processors** of many different kinds, including SMP, or symmetric multi-processing systems; massively parallel systems similar to that in FIG. 1 but having many more processors; or more loosely coupled networks of computers, such as networks of computer workstations.

DETD Similarly, many embodiments of the invention will not use the master and slave paradigm described above. Furthermore, in many embodiments of the invention the tasks described above as being performed on only **one processor** could be run on **multiple processors**. For example, the task of training non-terminal nodes

and using them to partition data for the training of leaf node neural networks should be parallelized if it will significantly increase the speed with which the tree can be built and trained. This would be the case if the number of non-terminal nodes becomes very large, or if the amount of computation associated with training each of them becomes large. For example, when the non-terminal nodes have hidden layers, as in FIG. 20, parallelization will tend to be more appropriate.

DETD It should be understood that in embodiments of the invention running on symmetric multiprocessing, or SMP, systems there will be no need to store a separate copy of the neural network tree for each processor, since all the processors will share a common memory, and there will be no need for **one processor** to transfer the records associated with a given leaf node to the processor which is going to train that leaf node, since they will be distributed to the processor that is going to train their associated leaf node when that fetches them from memory, itself.

DETD It should also be understood that, in some embodiments of the invention, neural tree networks similar to those shown in FIGS. 2 and 20 can be used to partition data for **multiple processors** which are using the data for purposes other than training hidden-layer neural networks. For example, such neural network trees can be used to partition data for **parallel processors** performing other types of modeling or analysis techniques, such as multi-dimensional statistical modeling, Kohonen networks, and discrimination trees. Similarly in some embodiments of the invention, the decision tree part of the entire neural tree network is replaced by another type of analytical classification algorithm, such as a Kohonen network, and the subsets of training data or apply data created by such a Kohonen network would be supplied to hidden layer neural networks. When used in a parallel environment the Kohonen network could be used to partition a training set into subsets, each representing classes of record.

DETD In other embodiments of the invention, a neural tree network of the type shown in FIGS. 2 and 20 could be applied in a process similar to that shown in FIG. 14, except that the partitioner 182, shown in FIG. 18, associated with the Apply Model object would pass records through the compressed representation of the decision tree part of the neural tree network, and the individual **parallel processors** receiving a partition of data set record sent to it by the tree partitioner would pass those records through the compressed representation of the corresponding hidden layer neural network. In such an embodiment, the decision tree partitioner would decide which of the processors executing the hidden layer neural networks a given record should be sent to, based on which of the decision tree's leaf nodes the record is routed to. If the system is running more than one hidden layer neural network on any processor node, the partitioner must label records sent to such nodes, indicating which leaf node the record has been associated with.

CLM What is claimed is:

1. A computer system comprising: P processors, where P is an integer greater than one; means for receiving a data set of data objects having N parameters, where N is an integer greater than one; means for **dividing** an N-**dimensional** data **space** having a separate dimension of each of said N parameters into M **sub-spaces**, each corresponding to a region of said N-dimensional space, where M is an integer greater than or equal to P, so each of said data set's data objects is located in one of said M **sub-spaces**, said means for dividing including means for dividing said space along boundaries which are non-orthogonal to said N dimensions; and means for associating different ones of said **sub**

-**spaces** with different ones of said processors, such that each of said P processors has a different set of one or more of said **sub-spaces** associated with it, including: means for distributing the sub-set of data objects located in each **sub-space** to the processor associated with that **sub-space**; and means for causing each processor to perform a computational process on each of the data objects so distributed to said processor.

2. A computer system comprising: P **parallel processors** , where P is greater than one; means for receiving a first data set of data objects to be processed; d-tree means including: means for storing a decision tree data structure having a plurality of non-terminal nodes, including a root node, and terminal nodes, wherein each of said non-terminal nodes has a plurality of child nodes, each of which is either one of said non-terminal or terminal nodes; means for storing a trainable decision criterion for each of said non-terminal nodes; and means for training said decision tree including: means for supplying a second set of said data objects to said root node as a training set, wherein said second set can either be equal to or different than said first set; and means for performing the following operation for each given non-terminal node in said tree; causing each given non-terminal node to use those of said training set data objects supplied to it to train said given node's decision criterion; and supplying each training set data object supplied to the given node to one of the given node's child nodes based on the application of the given node's decision criteria to the data object, once the given node's decision criteria has been trained; and means for using said decision tree to partition said first data set into at least M data sub-sets, where M is equal or greater than P; means for associating a different set of one or more of said data sub-sets with each of said P processors; means for distributing the data objects in each data sub-set to the processor associated with that sub-set; and means for causing each of said processors to perform a computational process on each of said data objects so distributed to the processor.

6. A computer system comprising: P **parallel processors** , where P is greater than one; means for receiving a first data set of data objects to be processed; d-tree means for using a decision tree to partition said first data set into at least M data sub-sets, where M is equal or greater than P; means for associating a different set of one or more of said data sub-sets with each of said P processors; means for distributing the data objects in each data sub-set to the processor associated with that sub-set; and means for causing each of said processors to perform a computational process on each of said data objects so distributed to the processor; wherein said d-tree means includes: means for storing a decision tree data structure having a plurality of non-terminal nodes, including a root node, and terminal nodes, wherein each of said non-terminal nodes has a plurality of child nodes, each of which is either one of said non-terminal or terminal nodes; means for storing a trainable decision criterion for each of said non-terminal nodes; and means for training said decision tree including: means for supplying a second set of said data objects to said root node as a training set, wherein said second set can either be equal to or different than said first set; and means for performing the following operation for each given non-terminal node in said tree; causing each given non-terminal node to use those of said training set data objects supplied to it to train said given node's decision criterion; supplying each training set data object supplied to the given node to one of the given node's child nodes based on the application of the given node's

decision criteria to the data object, once the given node's decision criteria has been trained; and wherein the decision criterion associated with one or more of said non-terminal nodes is a neural network and the decision criterion associated with at least one of said non-terminal node has a hidden layer.

8. A computer system comprising: P **parallel processors** , where P is greater than one; means for receiving a first data set of data objects to be processed; d-tree means for using a decision tree to partition said first data set into at least M data sub-sets, where M is equal or greater than P; means for associating a different set of one or more of said data sub-sets with each of said P processors; means for distributing the data objects in each data sub-set to the processor associated with that sub-set; and means for causing each of said processors to perform a computational process on each of said data objects so distributed to the processor; wherein said d-tree means includes: means for storing a decision tree data structure having a plurality of non-terminal nodes, including a root node, and terminal nodes, wherein each of said non-terminal nodes has a plurality of child nodes, each of which is either one of said non-terminal or terminal nodes; means for storing a trainable decision criterion for each of said non-terminal nodes; means for training said decision tree including: means for supplying a second set of said data objects to said root node as a training set, wherein said second set can either be equal to or different than said first set; and means for performing the following operation for each given non-terminal node in said tree; causing each given non-terminal node to use those of said training set data objects supplied to it to train said given node's decision criterion; supplying each training set data object supplied to the given node to one of the given node's child nodes based on the application of the given node's decision criteria to the data object, once the given node's decision criteria has been trained; and means for automatically setting the decision criteria of individual non-terminal nodes of the decision tree so as to achieve a desired ratio between the number of data objects supplied to each such node's child nodes; and means for automatically configuring the decision tree used so that it has P times I end nodes, where I is an integer, each of which end nodes defines one of said data sub-sets.

9. A computer system comprising: means for receiving a data set of data objects having N parameters associated with them, where N is an integer greater than one; means for **dividing** an N-**dimensional** data **space** having a separate dimension for each of said N parameters into M **sub-spaces**, each corresponding to a region of said N-dimensional space, where M is an integer greater than one, so each of said data set's data objects is located in one of said M **sub-spaces**; means for representing each of M hidden layer neural networks; means for associating each of the M **sub-spaces** with one of said M neural networks; and means for using the data objects in each of said M **sub-spaces** to train that **sub-space**'s associated hidden layer neural network.

12. A computerized method as in claim 11 wherein: the data objects have N parameters associated with them, where N is an integer greater than one; said using of training set data objects supplied to the given non-terminal node includes using said data objects to train the given non-terminal node's neural network to develop spatial criteria for **dividing** an N-**dimensional** data **space** having a separate dimension for each of said N parameters into a separate

sub-space for each of the given non-terminal node's child nodes, each of which **sub-spaces** corresponds to a region of said N-dimensional space, so that each of said data objects supplied to the given node is located in one of said **sub-spaces**; and said supplying of data objects to a one of a given node's child nodes is based on the given node's neural network's spatial criteria, such that data objects from different **sub-spaces** of said N-dimensional space are supplied to different ones of the given node's child nodes.

13. A computerized method including: receiving a first data set comprised of a plurality of data objects; and creating a decision tree data structure having a plurality of non-terminal nodes, including a root node, and terminal nodes, wherein each of said non-terminal nodes has a plurality of child nodes, each of which is either one of said non-terminal or terminal nodes, said creating of a decision tree including: creating for each non-terminal node a neural network; creating for each terminal node a neural network containing at least one hidden layer; supplying a second data set of data objects to said root node as a training set, wherein said second data set can either be equal to or different than said first data set; performing the following operation for each given non-terminal node in said tree; using the training set data objects supplied to the given non-terminal node to train the given node's neural network; and supplying each data object of said first and second data sets supplied to the given node to one of the given node's child nodes based on the output of the given node's neural network for the data object, once the given node's neural net has been trained; and using said data objects of the first data set supplied to a given terminal node to train the given terminal node's hidden layer neural network; wherein: the data objects have N parameters associated with them, where N is an integer greater than one; said using of training set data objects supplied to the given non-terminal node includes using said data objects to train the given non-terminal node's neural network to develop spatial criteria for **dividing** an N-**dimensional** data **space** having a separate dimension for each of said N parameters into a separate **sub-space** for each of the given non-terminal node's child nodes, each of which **sub-spaces** corresponds to a region of said N-dimensional space, so that each of said data objects supplied to the given node is located in one of said **sub-spaces**; said supplying of data objects to a one of a given node's child nodes is based on the given node's neural network's spatial criteria, such that data objects from different **sub-spaces** of said N-dimensional space are supplied to different ones of the given node's child nodes; said creating of a neural network for each non-terminal node includes creating a two layer neural network for such a non-terminal node, each of which has a plurality of inputs, no hidden layers, an output, and a series of weights between each input and said output, which weights define a vector in said N-dimensional space; and said supplying of each given one of a plurality of training set data objects to one of a given non-terminal node's child nodes is based on which side of an N-dimensional plan perpendicular to said vector in said N-dimensional space that a given data object is located.

14. A computerized method as in claim 11 wherein said using of training set data objects supplied to a given terminal node includes using such data objects supplied to each of a plurality of said terminal nodes to train said terminal node's associated hidden layer neural network on a different **parallel processor**.

15. A computerized method including: receiving a first data set comprised of a plurality of data objects; and creating a decision tree data structure having a plurality of non-terminal nodes, including a root node, and terminal nodes, wherein each of said non-terminal nodes has a plurality of child nodes, each of which is either one of said non-terminal or terminal nodes, said creating of a decision tree including: creating for each non-terminal node a neural network; creating for each terminal node a neural network containing at least one hidden layer; supplying a second data set of a data objects to said root node as a training set, wherein said second data set can either be equal to or different than said first data set; and performing the following operation for each given non-terminal node in said tree; using the training set data objects supplied to the given non-terminal node to train the given node's neural network; and supplying each data object of said first and second data sets supplied to the given node to one of the given node's child nodes based on the output of the given node's neural network for the data object, once the given node's neural net has been trained; using said data objects of the first data set supplied to a given terminal node to train the given terminal node's hidden layer neural network, including using such data objects supplied to each of a plurality of said terminal nodes to train said terminal node's associated hidden layer neural network on a different **parallel processor**; storing a copy of said decision tree, including the neural networks in its non-terminal and terminal nodes on each of a **plurality** of **processors**; dividing a set of data objects not in said training set into a plurality of data partitions, for each of said processors; and passing the data objects in each given processor's associated partition down the copy of the decision tree stored on the given processor, so each given **one** of said **processor**'s associated data objects is routed by the neural network in each of a succession of one or more non-terminal nodes to a respective child node, until the given data object is routed to a given terminal node in the processor's copy of the tree, after which the hidden layer neural network associated with said given terminal node is used to analyze the given data object.

20. A computer-implemented method for **parallel processing** of data, comprising: determining from the data at least one principle axis of the data; partitioning the data into a plurality of convex sets of data alone at least one plane orthogonal to each determined principle axis of the data corresponding to a number of processors; and in **parallel, processing** the convex sets of data using a plurality of analytic models on the processors, wherein each **processor** receives **one** of the convex sets of data and uses one of the plurality of analytical models.

21. A computer system method for **parallel processing** of a data, comprising: means for determining from the data at least one principle axis of the data; means for partitioning the data into a plurality of convex sets of data along at least one plane orthogonal to each determined principle axis of the data corresponding to a number of processors; and for processing the convex sets of data using a plurality of analytical models in parallel on the **plurality** of **processors**, wherein each **processor** receives **one** of the convex sets of data and uses of the plurality of analytical models.

L7 ANSWER 5 OF 6 USPATFULL
PI US 5864857 19990126

Mackly Monestime

SUMM The system of storing data in a hard disk unit in a format suitable for retrieval of multi-dimensional data involves a system in which data are linearly arranged in order of a nest of dimension coordinates in accordance with the memory array method of array data of a computer language and stored, a system in which the multi-dimensional data space is assumed to be divided in the direct product of **sub-spaces** in which data having effective values are distributed in sparsely and **sub-spaces** in which data having effective values are distributed densely and a memory area is allotted to only **sub-space** of the latter **sub-spaces** which is not empty to be pointed by the pointer array corresponding to the former **sub-space** (U.S. Pat. No. 05359724), and a grid file system in which attributes of record data of a relational model are considered as dimensions and the data **space** constituted by the **dimensions** is **divided** into rectangular parallelepipedic blocks so that pages for storing records belonging to the same block are allotted and indexes for recording the correspondence of coordinates of the blocks and the pages are prepared (Nievergelt, J., et al., "The Grid File: An Adaptable, Symmetric Multikey File Structure", ACM Transactions on Database Systems, vol. 9, No. 1, pp. 38-71, 1984).

SUMM The data retrieval can be made at a high speed by means of the **parallel processing**. In order that the **parallel processing** is effective, it is necessary to uniform loads imposed on a **plurality** of **processing** units. In order to uniform the loads of the data retrieval, it is important how data are allotted to the **plurality** of **processing** units and are stored. The relational database management system uses the round-robin partitioning, the hash partitioning and the range partitioning methods (DeWitt, D., et al., "The Gamma Database Machine Project", IEEE Transactions on Knowledge and Data Engineering, Vol. 2, No. 1, pp. 44-63, 1990).

SUMM In order to attain the high-speed retrieval of the multi-dimensional data, there can be considered the high-speed decision of a memory location of data to be retrieved, the high-speed apparent reading of data from a hard disk unit and the high-speed operation by application of the **parallel processing**. The high-speed reading of data is influenced by the reading efficiency of data per page and the page buffering efficiency and is further influenced by the clustering effect indicating how many pertinent data are clustered to be stored in the same page and the sparse data compression effect indicating how effective data distributed sparsely in the multi-dimensional data space are stored in the pages with reduced uselessness.

SUMM On the other hand, the data partitioning and storing system for **parallel processing** such as the round-robin partitioning, the hash partitioning and the range partitioning is to perform the simple cyclic allocation or the allocation using a single key value and does not utilize the characteristics of the multi-dimensional data usefully.

SUMM Further, in a **parallel processing** configuration using a **plurality** of **processing** nodes each including a CPU and a memory unit and interconnected through a communication network, multi-dimensional data are stored by a method comprising a first step of assigning a series of numerical coordinates to members constituting each dimension and dividing the coordinates by sections to thereby group the members and assign a series of numerical coordinates

to the obtained member groups, a second step of allotting to the memory units of the **plurality** of **processing** nodes, memory areas for page indexes constituted by entries corresponding to combinations relative to the plurality of dimensions, of the member groups determined in the first step and in which the entries are arranged in order of a nest with respect to a set of numerical coordinates of the corresponding member group, a third step of inputting data to be stored in the memory units of the **plurality** of **processing** nodes, a fourth step of determining block coordinates which are a set of numerical coordinates of a member group corresponding to the data inputted in the third step, from combinations of members of dimensions specifying the data, a fifth step of determining a processing node for storing the data inputted by the third step on the basis of the block coordinates determined by the fourth step so that data positioned in adjacent different diagonal planes of block coordinate space are assigned to different processing nodes, a sixth step of, in regard to the processing unit determined in the fifth step, calculating a distance from the head of the page indexes of the processing node determined in the fifth step on the basis of the block coordinates determined in the fourth step to thereby determine a memory location of a page index entry corresponding to the block coordinates and obtain the entry, a seventh step of, in regard to the processing unit determined in the fifth step, assigning a memory area of a page to the memory unit of the processing node determined in the fifth step and registering a page number of the page in the page index entry obtained in the sixth step when the page number is not registered in the page index entry, and an eighth step of, in regard to the processing unit determined in the fifth step, pairing the data inputted in the third step 3 and identification information of the data to store the paired data and identification information in the page having the page number registered in the page index entry obtained in the sixth step, and further the multi-dimensional data is retrieved by a method comprising a ninth step of inputting a retrieval condition represented by a combination of members of dimensions, a tenth step of deciding block coordinates corresponding to the retrieval condition inputted in the ninth step 9 from combinations of members of dimensions specifying retrieval data, an eleventh step of determining a processing node for storing the retrieval data having the block coordinates determined in the tenth step in accordance with an assignment method to the processing node of the memory data determined in the fifth step, a twelfth step of, in regard to the processing node determined in the eleventh step 11, making calculation of a distance from the head of the page index on the basis of the block coordinates determined in the tenth step to thereby determine a memory location of the page index entry corresponding to the block coordinates and obtain the entry, and a thirteenth step of, in regard to the processing node determined by the eleventh step, obtaining from the memory unit of the processing unit the page having the page number registered in the page index entry obtained in the twelfth step and obtaining data having identification information corresponding to the retrieval data from the page.

SUMM Further, according to the system, in the **parallel** **processing** configuration, data of blocks in adjacent diagonal planes in multi-dimensional data space are assigned to different processing nodes. In the multi-dimension retrieval, since data of parallel or vertical section are retrieved at the same time in dimensional axes, load on data retrieval is always uniform.

CLM What is claimed is:
9. A method of storing multi-dimensional data in a database management system including a **plurality** of **processing** nodes

each having a CPU and a memory unit and interconnected through a communication network and in which data identified by a combination of members of a plurality of dimensions are stored in said memory units of said **plurality** of **processing** nodes and a query relating to said data is processed, comprising: a step of assigning a series of numerical coordinates to members constituting each dimension and dividing said coordinates by sections to thereby group said members and assign a series of numerical coordinates to the obtained member groups; a step of allotting to said memory units of said **plurality** of **processing** nodes, memory areas for page indexes constituted by entries corresponding to combinations relative to said plurality of dimensions, of said member groups determined in said numerical coordinate assigning step and in which said entries are arranged in order of a nest with respect to a set of numerical coordinates of said corresponding member group; a step of inputting data to be stored in said memory units of said **plurality** of **processing** nodes; a step of determining block coordinates which are a set of numerical coordinates of a member group corresponding to said data inputted in said data inputting step, from combinations of members of dimensions specifying said data; a step of determining a processing node for storing said data inputted by said data inputting step on the basis of said block coordinates determined by said block coordinate determining step so that data positioned in adjacent different diagonal planes of block coordinate space are assigned to different processing nodes; a step of, in regard to said processing unit determined in said processing node determining step, calculating a distance from the head of said page indexes of said processing node determined in said processing node determining step on the basis of said block coordinates determined in said block coordinate determining step to thereby determine a memory location of a page index entry corresponding to said block coordinates and obtain said entry; a step of, in regard to said processing unit determined in said processing node determining step, assigning a memory area of a page to said memory unit of said processing node determined in said processing node determining step and registering a page number of said page in said page index entry obtained in said calculating step when the page number is not registered in said page index entry; and a step of, in regard to said processing unit determined in said processing unit determining step, pairing said data inputted in said data inputting step and identification information of said data to store said paired data and identification information in said page having said page number registered in said page index entry obtained in said calculating step.

10. A method of retrieving multi-dimensional data in a database management system including a **plurality** of **processing** nodes each having a CPU and a memory unit and interconnected through a communication network and in which data identified by a combination of members of a plurality of dimensions are stored in said memory units of said **plurality** of **processing** nodes and a query relating to said data is processed, comprising: a step of assigning a series of numerical coordinates to members constituting each dimension and dividing said coordinates by sections to thereby group said members and assign a series of numerical coordinates to the obtained member groups; a step of allotting to said memory units of said **plurality** of **processing** nodes, memory areas for page indexes constituted by entries corresponding to combinations relative to said plurality of dimensions, of said member groups determined in said numerical coordinate assigning step and in which said entries are arranged in order of a nest with respect to a set of numerical coordinates of said corresponding member group; a step of inputting data

to be stored in said memory units of said **plurality** of **processing** nodes; a step of determining block coordinates which are a set of numerical coordinates of a member group corresponding to said data inputted in said data inputting step, from combinations of members of dimensions specifying said data; a step of determining a processing node for storing said data inputted by said data inputting step on the basis of said block coordinates determined by said block coordinate determining step so that data positioned in adjacent different diagonal planes of block coordinate space are assigned to different processing nodes; a step of, in regard to said processing unit determined in said processing node determining step, calculating a distance from the head of said page indexes of said processing node determined in said processing node determining step on the basis of said block coordinates determined in said block coordinate determining step to thereby determine a memory location of a page index entry corresponding to said block coordinates and obtain said entry; a step of, in regard to said processing unit determined in said processing node determining step, assigning a memory area of a page to said memory unit of said processing node determined in said processing node determining step and registering a page number of said page in said page index entry obtained in said calculating step when the page number is not registered in said page index entry; and a step of, in regard to said processing unit determined in said processing unit determining step, pairing said data inputted in said data inputting step and identification information of said data to store said paired data and identification information in said page having said page number registered in said page index entry obtained in said calculating step; a step of inputting a retrieval condition represented by a combination of members of dimensions; a step of deciding block coordinates corresponding to said retrieval condition inputted in said retrieval condition inputting step from combinations of members of dimensions specifying retrieval data; a step of determining a processing node for storing said retrieval data having said block coordinates determined in said block coordinate deciding step in accordance with an assignment method to said processing node of the memory data determined in said processing node determining step; a step of, in regard to said processing node determined in said processing node determining step, making calculation of a distance from the head of said page index on the basis of said block coordinates determined in said block coordinate determining step to thereby determine a memory location of said page index entry corresponding to said block coordinates and obtain said entry; and a step of, in regard to said processing node determined by said processing node determining step, obtaining from said memory unit of said processing unit the page having the page number registered in said page index entry obtained in said calculation making step and obtaining data having identification information corresponding to the retrieval data from said page.

L7 ANSWER 6 OF 6 USPATFULL

PI US 5680525 19971021

DETD The process in the group of character expansion **processes** 21242 is **one** in which character contour line data is fetched from the character font database 2015 on the basis of the font type in the two-dimensional pattern attribute table 21245 and a character code constituting the character string stacked in the parameter stack memory 21247, and in which the fetched character contour line data is converted into character contour line data in a pattern domain or pattern definition space.

DETD FIG. 34 illustrates the object memory 3004 which is formed by the video graphic generation environment setting unit 3003. The data items of the

Mackly Monestime

subject element indicated at numeral 3023 are loaded by the database input unit 3002, and are stored in this object memory 3004. A route pointer 3041 for the Z buffer hidden-surface removal unit 3005 and a route pointer 3045 for the ray tracing unit 3007 are data items prepared beforehand, and they register the head addresses of a plurality of lists and the addresses of voxel sets to be stated below. The voxel sets indicated at numeral 3044 contain data for the ray tracing unit 3007, and they are the sets of voxel data which correspond respectively to **subspaces** (voxels) obtained by equally **dividing** a three-**dimensional space** where the whole subject exists. Each set of voxel data registers the list concerning the triangular plane which exists in the corresponding voxel. The voxel data items serve to efficiently process the computation of an intersection point. A list part 3042 is used by the Z buffer hidden-surface removal unit 3005, while a list part 3043 is used by the ray tracing unit 3007. The respective list parts 3042 and 3043 are independently managed by the route pointer 3041 for the Z buffer hidden-surface removal unit 3005 and the route pointer 3045 for the ray tracing unit 3007. Here, each of the list parts contains data for connecting the data items of the plurality of triangular planes, and it is composed of the address of the data of the triangular plane having been input (indicated by symbol .fwdarw. in FIG. 34) and that of the data of the triangular plane to be subsequently input. In the case of FIG. 34, then the shape data of one triangular plane has been input, the lists for both the units 3005 and 3007 are formed in the object memory 3004. In this Manner, the addresses concerning the shape of the subject are managed independently by the list part 3042 for the Z buffer hidden-surface removal unit 3005 and by the list part 3043 for the ray tracing unit 3007, whereby the processes of the Z buffer hidden-surface removal and the ray tracing can be simultaneously performed.

=>